Learning Jazz Grammars

Jonathan Gillick (Wesleyan University, Connecticut, USA)

Kevin Tang (Cornell University, New York, USA)

Robert Keller (Harvey Mudd College, California, USA)

Supported in part by National Science Foundation REU.
Jazz Learning Approaches

- **Traditional:**
  
  Transcribe, and learn to imitate, solos of great masters.

- **Impro-Visor:**
  
  Write your own solos, understanding the chord changes in the process.
  
  Use part or none of what you wrote, as you please, in actual improvisation.
Background: Impro-Visor

- **Improvisation + Advisor**

- Educational software to help musicians:

  - teach themselves to improvise jazz over chord changes

- “Advice” includes suggestions for jazz licks:

  - From a user-modifiable database, or

  - From a “lick generator”.
Impro-Visor v.4 Lead-sheet Screen Shot

12-Bar Blues

Style: swing
F13

Bb13 Bb7 F13 Cm9 F13b9

Gm9 C13b9 F13 D7#5#9

Bb13 Bb7 F13 D7#5#9

Gm9 C13b9
Constructing a Lick Database

- Educational in its own right
- Laborious and time-consuming
- Difficult to achieve coverage of every situation (e.g. every possible pair of consecutive chords)
- That motivated the Lick Generator.
Lick Generator

Chord Sequence In

Gm9 | C13b9

Generate

Lick Out

Gm9

C13b9
Lick Generator (SMC 2007, Lefkada)

- Built on a **probabilistic context-free grammar**.
  
  \[
  \begin{align*}
  V_2 &\rightarrow X_2 \\
  V_2 &\rightarrow X_8 \ X_4 \ X_8 \\
  V_4 &\rightarrow X_4
  \end{align*}
  \]
  
  \( \text{prob} \ 0.2 \) \( \text{prob} \ 0.8 \) \( \text{prob} \ 1 \)

- Some rules add arguments to control **filling beats**
  
  \[
  \begin{align*}
  P(Y) &\rightarrow V_2 \ P(Y-2) \\
  P(Y) &\rightarrow V_4 \ V_4 \ P(Y-2) \\
  P(0) &\rightarrow ()
  \end{align*}
  \]
  
  \( \text{prob} \ 0.5 \) \( \text{prob} \ 0.5 \) \( \text{prob} \ 1 \)

- \( P(N) \) is the **start symbol** if there are \( N \) beats to fill

- **User-modifiable**
  
  (by a sophisticated user)
Constructing a Grammar
Difficulties Analogous to Constructing a Database

- Also educational in its own right
- Trial-and-error to get exactly the right style
- Motivates Grammar Learning
Grammar Generator

Performances Transcriptions In

Learn

Grammar Out

(rule (V2) (S8 S8 S8 S8) 0.3)
(rule (V4) (H8/3 H8/3 A8/3) 0.01)
(rule (V4) (H8/3 H8/3 H8/3) 0.05)
(rule (V4) (H8/3 S8/3 H8/3) 0.02)
(rule (V4) (N4) 0.22)
(rule (V4) (V8 V8) 0.72)
(rule (V8) (H16 A16) 0.01)
Vision

Grammar generator would help construct a *repertoire* of selectable grammars

- Different performance styles
- Varying difficulties
- Lick-trading patterns
Grammar Learning Problem

- Create, from a corpus of transcribed performances, a grammar that captures the stylistic elements represented in the corpus.

- How do we capture the style?
  - Note categories form abstract melodies
  - Contours, slopes give more micro-structure
  - Clustering reduces complexity
  - Markov chains give macro-structure
Note Categories

Over a given chord, each note in the chromatic scale has a pre-specified category, one of:

- Chord tone
- Color tone (not in chord, but sonorous with it)
- Approach tone (frequently used in jazz)
- Other (“outside”, in jazz parlance)
Color-Coding Note Categories

- Chord tone: Black
- Color tone: Green
- Approach tone: Blue
- Other: Red
Coloration Examples

C Major 7th chord

C minor 7th chord
Textual Representation of Melody

Concrete melody: r8 b8 e+8 d+8 b8 f8 bb8 b8
Textual Representation of Note Categories

- **Chord tone:** Black \( C \)
- **Color tone:** Green \( L \)
- **Approach tone:** Blue \( A \)
- **Other:** Red
  - Scale: \( S \)
  - Arbitrary: \( X \)
- **Abbreviation:** \( H = C \) or \( L \)
- **R = rest**
Textual Representation of Abstract Melody

Concrete melody: r8 b8 e+8 d+8 b8 f8 bb8 b8

Abstract melody: R8 L8 C8 L8  C8 S8 A8 C8
Many-Many Relations

One
abstract melody

Many
concrete melodies
Many-Many Relations

One concrete melody

Many abstract melodies

But this direction can be turned into a function using a few procedural rules.
Many-Many Relations

One concrete melody \rightarrow One abstract melody
rules
Our Generation Principle

One concrete melody → Learn: One abstract melody

rules

Generate:
Many concrete melodies
First Challenge

- Abstract melodies, as permitted so far, are too liberal in their generation.

- They cannot capture melodic contour.
Solution to First Challenge

- Add new units to represent slope & contour

- e.g. unit (slope 1 5 S8 S8 C4)
  
  minimum rise

  maximum rise

  note categories
Solution to First Challenge, cont’d

- Chained slopes = Contours

\[(\text{slope 1 5 S8 S8 C4})(\text{slope -2 -4 L4})(\text{slope 5 5 C4})\]

[Some touching up may be done if the constraints of the contour can’t be met exactly.]
Second Challenge

- Retaining too many abstract melodies in the grammar is apt to slow things down.
Solution to the Second Challenge

- **Cluster** the abstract melodies and select a few representatives of each cluster.

- This entails creating a distance metric between abstract melodies, which we have done.

- So far, only k-means clustering has been used.
Parameters Used in Metric

- number of notes
- total duration of rests
- average maximum slope
- phrase starts on or off the beat
- order of the contour (number of direction changes)
Cluster Examples

Three representatives from the same cluster:
Third Challenge

- Concatenating abstract melodies arbitrarily does not provide for a convincing style emulation.
Solution to the Third Challenge

- Infer transition probabilities for **Markov chains** between clusters.

- States of Markov Chain are non-terminals in the grammar.

- Incorporate the probabilities into the grammar rules, which are already probabilistic.
Markov Imitation Example

Original two-measure melody

and two measures of imitation with a Markov model:
Markov Models

Markov model diagram for grammar rules (next slide):
Example of Inferred Markov Grammar Rules

Inferred from a corpus of Charlie Parker solos:

- (rule (START Z) ((Cluster0 Z)) 0.23)
- (rule (START Z) ((Cluster1 Z)) 0.25)
- (rule (START Z) ((Cluster2 Z)) 0.52)
- (base (Cluster0 0) () 1)
- (base (Cluster1 0) () 1)
- (base (Cluster2 0) () 1)
- (rule (Cluster0 Z) (Q0 (Cluster2 (- Z 1))) 52.00)
- (rule (Cluster0 Z) (Q0 (Cluster1 (- Z 1))) 24.00)
- (rule (Cluster0 Z) (Q0 (Cluster0 (- Z 1))) 24.00)
- (rule (Cluster1 Z) (Q1 (Cluster2 (- Z 1))) 54.04)
- (rule (Cluster1 Z) (Q1 (Cluster1 (- Z 1))) 27.95)
- (rule (Cluster1 Z) (Q1 (Cluster0 (- Z 1))) 18.01)
- (rule (Cluster2 Z) (Q2 (Cluster2 (- Z 1))) 50.75)
- (rule (Cluster2 Z) (Q2 (Cluster0 (- Z 1))) 25.53)
- (rule (Cluster2 Z) (Q2 (Cluster1 (- Z 1))) 23.72)
Generated Solo Example

Now's The Time
Charlie Parker

Chorus 1
Improvisation

\[ \text{F7} \quad \text{Bb7} \quad \text{Bb7} \quad \text{Bo7} \quad \text{F7} \quad \text{F7} \quad \text{Cm7} \quad \text{F7} \quad \text{F7} \quad \text{Bb7} \quad \text{Bo7} \quad \text{F7} \quad \text{D7alt} \quad \text{D7alt} \quad \text{Gm7} \quad \text{C7} \quad \text{C7} \]
Assessment Experiment

Using our methodology, we inferred grammars from Clifford Brown, Freddie Hubbard, Miles Davis.

Generated solos from each grammar on a different song

Asked users to identify which solo belongs to which artist, how closely they resembled the style of the original

Used same midi instruments, same song that was generated on, and hidden artist names as control elements
Results

85% correct for matching all three generated solos to artists

Almost all subjects selected either “somewhat close” or “quite close” for resemblance between solo and original

Good diversity in musical knowledge of subject sample space
Impro-Visor v.4 Grammar Learning Tool

**Grammar Learning**

Please follow these steps to learn a new grammar from a corpus of solos as a folder of leadsheets. Click the rectangular buttons from top to bottom.

**Step 1:** Load the grammar on which you wish to build, such as Bare.grammar. If you do nothing, it will build on whatever grammar is current.

**Step 2:** IMPORTANT: Use **Save as ...** in the Grammar menu to save your new grammar under a new name, in case you want to return to the old grammar. Also save your leadsheet if you need it, as the leadsheet window will be used as a workspace.

**Step 3:** (Optional) Set the parameters below:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window Size (beats)</td>
<td>4</td>
</tr>
<tr>
<td>Window Slide (beats)</td>
<td>2</td>
</tr>
<tr>
<td>Number of Representatives per Cluster</td>
<td>12</td>
</tr>
<tr>
<td>Use Markov (ordered connection of phrases) Chain length</td>
<td>3</td>
</tr>
</tbody>
</table>

**Step 4:** Clear the accumulation, unless you want to accumulate from several corpuses into the same grammar.

**Step 5:** Select a corpus of solos from which to learn. Each solo is a leadsheet file. Selecting any file any a folder is equivalent to selecting the entire folder. The leadsheet you selected will be left in the window at the end. The process is over when the last chorus appears.

**Step 6:** Click the **Add Accumulation** button to create and save the grammar and Soloist file.

You can try your grammar at generation immediately without further loading, on the current or any other leadsheet, however it will not appear in the main window until you restart the program.

**Step 7:** Try Generating a Solo with Your Grammar
Conclusions

- Devised a method of grammatical inference for jazz solo generation

- Key Ideas
  - Abstract musical representation
  - Extraction windows
  - Clustering/Markov Models

- Successful implementation of grammatical inference and solo generation for a variety of artists
For Further Information

- [ ] www.impro-visor.com
  (don’t forget the hyphen)

- [ ] or Google impro-visor

- [ ] launch.groups.yahoo.com/group/impro-visor/
  to download our software (free!)